Explanation-Oriented Classification via Subspace Partitioning

Madalina Fiterau* mfiterau@cs.cmu.edu
Artur Dubrawski† awd@cs.cmu.edu

October 16, 2012

Abstract

In many applications of predictive analytics the ability to explain the context of predictions is often as important as their accuracy. This paper introduces Explanation-Oriented Partitioning (EOP), a method that uses a small number of low-dimensional projections of data, each with its own discriminator, to find explainable classifications. This meta-algorithm can work with various base classifiers, leveraging local performance of the discriminator to identify low-dimensional regions of the feature space where data is easily separable. EOP picks out multiple clusters of well-classified points from informative projections of data, maintaining compactness of the resulting models. EOP can also identify regions of the feature space where data is particularly noisy. Compared to other methods, EOP models are accurate at low complexities, requiring fewer projections to achieve a user-specified accuracy threshold. Our experiments show how EOP finds succinct descriptions that capture patterns in data, providing this essential capability of predictive systems designed to work in close interaction with human users.

1 Introduction

The typical design methodology of data-driven analytic systems focuses on optimization of an empirical loss function appropriate for the task. In predictive scenarios, such as classification or regression, considerable efforts are often spent on model selection and tuning so that quantitative metrics of accuracy and reliability of the solution are optimized. Similarly, the focus of descriptive is maximizing the fidelity with which the underlying data mechanisms are represented, while minimizing the risk of overfitting. The emphasis on accuracy of the resulting models often supersedes the desire for explainability. This can be seen among popular out-of-the-box high performers such as boosting or random forests, which often appear to the user as black-box oracles with very little to offer in terms of explaining their predictions. We acknowledge the practical need for such methods. We observe that in many field deployment scenarios, especially in the context of data mining, the end users require explainable, transparent-box models, which also maintain reasonable accuracies.

We introduce Explanation-Oriented Partitioning (EOP), a method that is designed for this task. It uses a few low-dimensional projections of data, each with its own discriminator, to learn explainable classifications. This meta-algorithm can work with any type of discriminators - SVMs, logistic regression, random forests. It leverages local performance of the classifier to identify low-dimensional regions of the feature space where data is well-classifiable. The individual result for a query point includes concise information about the region of feature space which supports the current prediction, in addition to the label of the most likely class. EOP can also identify regions of the feature space where data is noisy (difficult to classify). EOP is an iterative algorithm. In the first iteration, it selects the one projection that is most effective in support of the classification task, among all data projections of a given dimensionality. The data that cannot be accurately classified and explained using the current model becomes the focus of the next iteration. This way we obtain a hierarchical sequence of models similar to a decision list, each component of which is a projection of data that can be used to classify and explain the partition assigned to it accurately and without excessive complexity.

2 Related Work

The hierarchical flavor of EOP makes it somewhat similar to boosting [11]. The way it splits the data brings up a reference to partitioning models such as CART [4]. Empirical evaluation shows that, compared to boosting, EOP produces clear-box and often more compact models at the price of a slight loss in classification accuracy. Compared to CART, EOP models tend to be more accurate at low complexities, .

EOP is also an ensemble model. Ensemble learning has been long known to enable great improvements of model accuracy by combining the capabilities of mul-

*Machine Learning Department, Carnegie Mellon University
†Robotics Institute, Carnegie Mellon University

tiple base classifiers [25, 1, 14, 21]. Methods such as Winnow [15] and Boosting [10] are guaranteed to decrease training error with each iteration by tweaking either the voting coefficients or the weights of the training data records. Performance can be further enhanced by combining those techniques [23]. However, there has been much debate as to what extent the accuracy of prediction is indicative of the ability of an algorithm to uncover the processes behind data [3]. To answer that, approaches that simplify trained ensembles have been proposed [8, 7], and methods that replace accurate black-box models with more interpretable equivalents [6, 16]. Other techniques attempt to improve understandability by simplifying or compressing the feature space [13, 24, 20]. So far only a handful of algorithms have been specifically designed to yield understandable models. However, rules learned in [18] can be hard to visualize, and itemset mining [17] is not quite native for classification tasks. In Feating [22], submodel selection relies on simple attribute splits followed by fitting local predictors. EOP reverses this sequence: it first identifies useful discriminators, then builds decisions (more generic than those of Feating).

EOP uses intuitive hierarchical models like decision lists or trees, while also specifically addressing the objective of maintaining a specified minimal accuracy and using as compact representation as possible. Alternative approaches train potentially highly-accurate black-box classifiers followed by post-processing of the results to extract insights on its decisions. EOP replaces this often complicated two-stage process. We empirically compare EOP to a representative group of the methods listed above. Our experiments show EOP finds succinct descriptions that capture patterns in data. This ability is a crucial aspect of practical utility of prediction systems working in close interaction with human users.

3 Explanation-Oriented Partitioning

3.1 EOP Learning The Explanation-Oriented Partitioning algorithm (EOP) iteratively selects projections of data in which the data can be classified with high accuracy. EOP can use any externally supported classifier - in the experiments shown below we use Support Vector Machines [5] and decision stumps. In each of the selected projections, EOP identifies contiguous areas (regions) in which predictions are consistently accurate. These regions are then used to explain predictions made for data inside their bounds. An example prediction produced by the trained EOP model in response to a test data query could, for instance, pronounce: *'This query appears to belong to class A. It can be shown that in the scatterplot of data projected onto x_1 and x_7 this query is densely surrounded by instances that belong to the same class in the area bounded by* $13.5 < x_1 < 25.0$ *and* $0.4 < x_7 < 1.3$*'*. The user thus obtains the context of the prediction and the ability to confirm its validity.

The algorithm employs a few parameters. The users specify the target classification error rate ϵ, the regularization parameter for the trained classifiers λ - its meaning is obviously specific to the chosen classifier type, and η used to control complexity of the projections of data. The users supply the training data and the algorithm exhaustively evaluates all feasible projections of a selected dimensionality - we use 2-dimensional projections in the experiments considered below. EOP then identifies the projection π which allows for the most accurate classification of data given the particular settings of parameters (λ, η), and the corresponding trained classifier h. The next step is to identify regions in the current projection where the data is predominantly correctly classified, with the maximum within-region classification error rate of ϵ. There may be multiple such potentially overlapping regions in any of the considered projections. EOP uses a distinct subset of data to calibrate identified regions by expanding or contracting their boundaries, even deleting some of them, to prevent overfitting. Finally, the training data captured by the calibrated regions is removed from consideration, and the remainder becomes the input for the next EOP iteration.

The resulting model is therefore a hierarchy of projections with corresponding classifiers and regions selected in these projections. When the EOP model is queried with a test data point, the top component of the hierarchy is inspected first. If the query falls inside any of the regions associated with this sub-model, its classifier predicts the class label, and it is returned to the user together with the description of the invoked region. Otherwise, the algorithm falls back to the next component of the hierarchy. Therefore, the query-time operation is that of a decision list.

The basic stopping criterion for EOP's learning procedure is the exhaustion of training data. It may not be attainable if the required accuracy of classification in a region, ϵ, is overly restrictive. If so, EOP can either dynamically relax ϵ until all training data is accounted for or it can leave a certain amount of hard to handle data unresolved. The choice depends on the application.

3.2 Implementations of EOP Region Finding and Calibration So far we have described the high-level EOP algorithm without providing specifics regarding region extraction. We have mentioned that EOP is flexible regarding the choice of the base classifier. It can also rely on various approaches of region extraction. Below we detail two such methods, a parametric and a non-parametric segmentation of data, outlining the trade-offs that come with each of them and how

bootstrapping can help in making the process robust.

3.2.1 Bounding Polyhedra One way of characterizing regions of consistently classifiable data is to encase them in simple boundaries, such as polyhedra. For a polyhedron to qualify as a region of interest, the fraction of misclassified data in it should not exceed ϵ. Given a particular projection, the task of finding a polyhedron that maximizes data coverage while satisfying the ϵ condition is NP-hard. Instead, we use simple heuristics to achieve satisfactory results.

We used a method which starts with a randomly selected correctly classified data point as a seed, and subsequently adds more such points located nearby, growing the region for as long as the minimum accuracy constraint can be maintained. After such a set is found, the process is restarted with another correctly classified data point that is not yet enclosed in any of the previously constructed polyhedra.

The process ends when all the correctly classified data is consumed. The algorithm allows the resulting polyhedra to overlap. In order to prevent overfitting, EOP calibrates the result using hold-out data. Polyhedra that do not include any of the calibration data or for which the mean error over the enclosed calibration data exceeds ϵ are deemed unreliable and removed. The remaining regions are subject to shape adjustments - they can shrink to exclude noisy data or expand to include clean validation data located near the region boundary.

The EOP learning algorithm takes into account the complexity of the set of polyhedra that survive calibration. We estimate complexity as the sum of the number of facets of each polyhedron across all polyhedra associated with the particular classifier. Note that the presented process can be easily tailored to search for simplexes or hyper-rectangles. The latter are especially attractive from the potential end-user perspective if we require their sides to align with the axes of the data coordinate system. The resulting region boundaries can then be expressed using highly intuitive interval queries. Also, geometric boundaries of regions do not have to be linear. Elliptical bounds can be used as well.

3.2.2 Nonparametric Regions Presenting patterns in a parametric form is, although intuitive, not the only way to express conditions on the involved data points. An alternative is to estimate the density of the correctly/incorrectly classified data and define the region using a threshold on the likelihood ratio. A higher ratio pinpoints data that is easily classifiable and therefore eligible for inclusion in one of the reported regions. We go around the need to estimate densities and instead score each candidate data point using the information of the distances to its correctly and incorrectly classified neighbors. The number of neighbors considered is $k = \frac{1}{\epsilon}$. The intuition is that if the point being scored is a part of a contiguous region that satisfies the required accuracy, it should not have more than one incorrectly classified point within its k-neighborhood. This property can be used for pruning data unfit for inclusion in any of the regions worth reporting, and bounding the search for computation time savings.

We compute a weight for each of k neighbors of point p: its i^{th} neighbor n_i is assigned a weight of $w_i = \frac{1}{1+d(p,n_i)}$, where $d(p, n_i)$ denotes the distance between p and n_i. The score is then computed as the ratio of the sum of weights of the correctly classified neighbors to the sum of weights of all k neighbors:

$$Score(p) = \frac{\sum_{i=1}^{k} \frac{1}{1+d(p,n_i)} C(n_i)}{\sum_{i=1}^{k} \frac{1}{1+d(p,n_i)}}$$

$C(n_i) = 1$ if n_i is correctly classified, 0 otherwise.

We compute the scores for the complete set of training data. The next step is to determine the threshold of the score to decide which of the correctly classified data points should be included in the reported region. We identify a subset S_g of data with scores greater than $1 - \epsilon$. The region eligibility threshold is then set as the lower of $1-\epsilon$ and $\frac{|S_g \cap S_c|}{|S_g|}$ where S_c is the subset of correctly classified training data. Similarly to the parametric approach, we can use calibration data to adjust the eligibility threshold in order to robustify the nonparametric EOP regions against overfitting.

3.3 EOP Operation

3.3.1 Example using Synthetic Data Let us consider a simple example to illustrate EOP operation. We synthesized a data set with 3-dimensional continuous input space and a binary output. The data belonging to the first class - depicted in red in the graphs below - follow a uniform distribution [0,5] over all 3 features. The points in the second class - shown in blue - have been generated using two models each composed of a bivariate Gaussian and a uni-variate uniform distribution. One of the models is Gaussian w.r.t. features 1 and 2, and uniform in dimension 3, while the other is Gaussian for features 1 and 3, and uniform in feature 2. Figure 1 shows this data projected on all pairs of features.

Each row of graphs in Figure 2 illustrates one iteration of the EOP algorithm. For each row, the graph on the left represents the scatterplot of data considered at that iteration - points belonging to different classes are shown with distinct symbols: '+' for label 1 and 'o' for label 0. The center graph represents the probability of accurate classification computed for each data point computed using the k-nn score, shown as a colormap;

the red side of the spectrum denoting the points on which the classifier will do well, while the blue end shows points where correct classification is unlikely. Finally, the graph on the right depicts the outlines of the selected regions. *Note that this representation is different from what the EOP users are shown for processed queries.* Query resolutions are presented in Section 4.3.

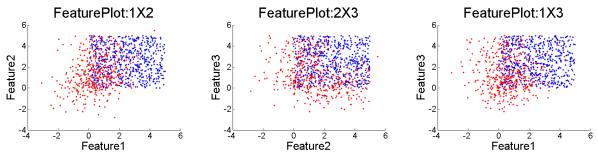


Figure 1: 2-D projections of synthetic data

The first iteration of EOP selects the projection of data onto features 1 and 3 as enabling the most accurate classification overall. The top left graph in Figure 2 shows the training data in this projection with the color and symbol-coded class labels. A classifier h_1 is trained on this 2-D problem. The data that h_1 classifies correctly are marked with '+' in the central graph, while the misclassified data points are marked with 'o'. The color intensity of these symbols notifies the proximity to neighboring correctly classified training examples. As expected, the classification is more confident at the farther sides of the classification boundary, and not so convincing wherever the training data shows significant overlap of the two classes. The top right graph encapsulates data considered sufficiently explainable by the above described nonparametric procedure to be included in region R_1. In the prediction phase, any data point belonging to R_1 will be classified using h_1.

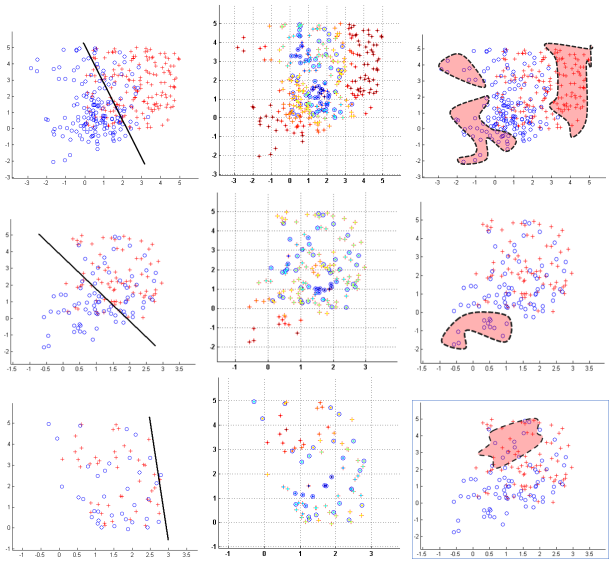


Figure 2: The first 3 iterations of nonparametric EOP executed on synthetic data set - one row per iteration. Depicts EOP's internal workings, not shown to users.

The training data in R_1 are eliminated from consideration in subsequent iterations; EOP the identifies regions R2 and R3 in projections [1,2] and [1,3] respectively - the second and third rows of Figure 2.

When EOP receives a query, it first projects it on features [1,3]. If the point belongs to R_1, it is classified with h_1; if not, it is projected on [1,2] and if it belongs to R_2 it is classified with h_2. Otherwise it is passed on to the following projection and so on. If the point does not fit into any of the learned regions, it can be either left unclassified or assigned the most common label.

Parametric EOP using rectangular regions was also ran on this data. Figure 3 shows all of the selected rectangular regions for $\epsilon = 0.1$ (left) and the regions that survive pruning with calibration data (right). This example illustrates the importance of using calibration data in preventing complexity and overfitting. Calibration of models using a separate subset of training data is not unusual in the practice of machine learning.

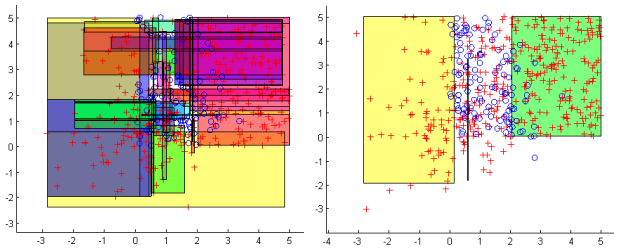


Figure 3: Selection of consistently classifiable regions and results of pruning shown using the synthetic data.

3.3.2 Avoiding unnecesary complexity Let us consider a variation of the classic XOR problem: a two-dimensional binary data consisting of four separable regions symmetrically distributed as shown in the top left plot of Figure 4. Decision trees are known to have difficulties with such data because their learning algorithms follow greedy strategies of maximizing immediate gains at each step of tree development. In our example, CART ends up chunking the data into many small slices, instead of discovering a visually obvious geometric pattern of data distribution, as depicted in the top right picture in Figure 4. A regularized model obtained with CART consists of 21 nodes - substantially more complex than the theoretically optimal model with only 3 nodes.

EOP fares substantially better. It starts by training the best linear separator of the complete distribution of data, which is equivalent to a default classifier and it classifies all data as belonging to the more populous class 1. The training samples of class 1 are then marked as correctly classified, and two regions of highly reliable classification are identified as shown in the bottom left graph of Figure 4. In the next iteration, EOP identifies regions covered by the points of class 0 as shown in the

bottom right of Figure 4. The hierarchy of the resulting model will have 3 levels: one for the regions of class 1, one for the regions of class 0, and a third level for noisy data points located close to the boundary. EOP will either not commit to classifying them, or label them as members of the most frequent class.

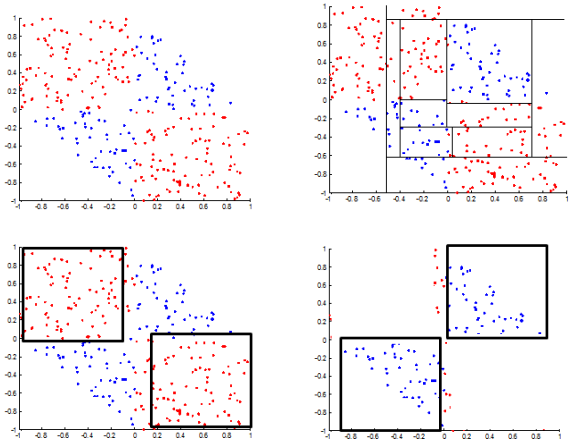


Figure 4: XOR data (top,left). Corresponding decisions learned by CART (top,right). First set of regions found by EOP (bottom,left). Second set of regions found by EOP (bottom, right).

4 Experimental Evaluation

The previous examples provide intuition for how EOP can find concise patterns in data. We now focus on quantitatively comparing the performance of EOP versus classical alternatives, AdaBoost and CART, as well as a handful of more contemporary algorithms (random forests [2], multiboosting [23], subspacing [13], and feating [22]), using realistically complex synthetic and real-world data. The results indicate that EOP is comparable in terms of the attainable classification accuracy to these alternatives. However, EOP achieves it using simpler models and more informative initial projections with easily classifiable subsets of data.

We used synthetic data and multiple real world datasets drawn from the UCI repository [9]. The two-class synthetic data was generated by sampling from uniform distributions along all coordinates. Then, we injected additional data drawn from randomly shaped and positioned Gaussians - each containing points from only one class - spanning a low number of randomly selected dimensions. Each synthetic set included 7 such injections - one 4-dimensional, two 3-dimensional and four 2-dimensional. The data is more difficult if the patterns overlap to a greater extent. All statistical tests are made using 10 datasets, each with 10 real-valued input features and 3,000 data points. In UCI datasets, we ignored any non-numeric input features, if present. We used *Breast Cancer Wisconsin* (10 inputs, 569 records), *MiniBoone* (10 inputs, 5000 records), *Breast Tissue* (9 inputs, 1696 records), and *Vowel* (10 inputs, 990 records) data. Two thirds of each were available for training models, the rest was used for testing, and only the test set results are presented below.

We successfully executed EOP on larger datasets - up to thousands of features - by replacing the standard exhaustive approaches with projection selection and region identification with randomized sampling.

4.1 Accuracy and Complexity

4.1.1 Comparison to Boosting Adaboost trains a sequence of weak classifiers by increasing at each iteration the weight of the points that were incorrectly classified at previous iterations. The training set error decreases at each step, and is theoretically guaranteed to go to zero in the limit, after a sufficient number of iterations. In practice, test set error often reaches a small non-zero plateau. EOP is similar to Adaboost in its iterative sub-setting of data into increasingly difficult subproblems. It also follows the basic principle: no matter how low-performing a particular classifier is, it will do well in some part of the feature space. However, the intents differ: boosting primarily tries to lower error rates, while EOP prioritizes explainability of the models, while trying to maintain decent accuracy.

Table 1: Comparison of nonparametric EOP (E) and boosting (B) - both with SVM base classifiers - in terms of accuracy (A) and complexity (C) on artificial data

	B A	***B C***	***E A***	***E C***
DS1	0.97	48	0.964	22.53
DS2	0.904	63	0.903	25.5
DS3	0.97	217	0.964	39.12
DS4	0.928	39	0.922	28.21
DS5	0.944	97	0.928	28.97
DS6	0.918	149	0.931	59.87
DS7	0.954	206	0.964	27.63
DS8	0.968	214	0.967	23.08
DS9	0.978	9	0.976	27.67
DS10	0.914	138	0.895	41.45
Mean	0.9448	118	0.941	32.403
Stdev	0.027	77.896	0.029	11.489
T-test	0.832	0.003		

Table 1 presents the outcome of running nonparametric EOP and Adaboost on synthetic data. Both EOP and Boosting use SVMs as base classifiers. The error is low in both cases, and although boosting beats EOP on average by a small margin, the difference is not systematic as indicated by the p-value of the paired T-test. On the other hand, EOP outperforms boosting in simplicity and the difference is systematic. We use operational complexity as the metric of reference - the expected number of operations to be performed when a

test data point is classified. For boosting, this means a vector multiplication for each classifier - the number of iterations picked by cross validation; for nonparametric EOP it is the expected number of neighbors that need to be taken into account before a decision is reached plus the classification effort - one vector multiplication.

4.1.2 Comparison to CART A prototypical white box method, CART, learns decision trees for classification by splitting the feature space into regions that have consistent values of the output labels. Heavy pruning and cross-validation are used to prevent overfitting. The outcome of the algorithm is meaningful as well as accurate. Additionally, because the feature space is split only as necessary, the resulting model can be compact.

EOP groups and filters data by how well a classifier can deal with it, so the assignment of the output class is indirect - the label is assigned by the corresponding classifier. Also, EOP splits data differently than CART. Its sequential approach leads to a list hierarchy rather than a tree, often yielding lower complexity; nonetheless it is subject to similar dangers of overfitting. Since both methods produce human understandable models, a clear-box comparison is drawn. For fairness, we compare EOP models that use decision stumps as base classifiers - better classifiers will yield higher accuracies.

Table 2: Comparison of accuracy and model complexity obtained by different methods - Random Forests (RF), Multiboosting (Mb), Subspacing (Ss), Feating (FT), CART, nonparametric EOP (N-EOP) and parametric EOP (R-EOP) - on datasets from the UCI repository.

Acc	**RF**	**Mb**	**Ss**	**FT**	**CART**	**N-EOP**	**R-EOP**
BCW	0.942	0.922	0.912	0.938	0.908	0.905	0.894
MB	0.86	0.849	0.87	0.728	0.856	0.83	0.83
BT	1	0.943	1	0.956	1	0.982	0.78
Vow	0.944	0.841	0.899	0.868	0.947	0.872	0.842
Comp	**RF**	**Mb**	**Ss**	**FT**	**CART**	**N-EOP**	**R-EOP**
BCW	325	15	30	20	3	3	2
MB	2456	60	30	20	19	8	7
BT	18	15	30	20	15	4	2
Vow	516	60	30	20	31	8	4

We compared EOP to CART as well as a few other contemporary and relevant algorithms on real data from the UCI repository. Random Forests [2] is a popular and and highly-competitive black-box technique that learns a bagged ensemble of decision trees. Random Subspacing [13] learns a random forest by sampling a subset of features to train each tree. Multiboosting [23] bridges the gap between ensemble learning methods designed to reduce the bias component of the predictive error (e.g. boosting) with those that take on variance (e.g. bagging). Feating (feature-subspace aggregation) [22] is a relatively recent method that splits the data space through a decision tree and trains local models.

Table 2 summarizes the comparison. Although nonparametric EOP does not come first on accuracy, it typically outperforms one or two counterparts. However, in most cases it offers a substantial reduction in complexity - only CART matches nonparametric EOP on Breast Cancer data. Parametric EOP allows further savings of complexity at the expense of slight reduction of accuracy. Figure 5 compares accuracy of nonparametric EOP and CART computed during learning the structure, at subsequent levels of the respective hierarchies. EOP achieves better performance for all datasets at the first level of hierarchy and for most of them at the second level. CART requires deeper structures to finally take the lead at the cost of additional complexity.

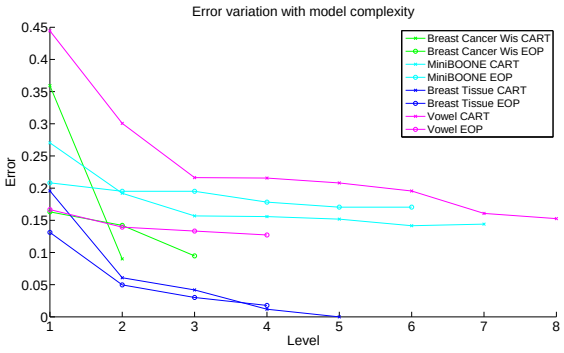


Figure 5: Variation of error with model complexity for CART(+) and nonparametric EOP(o) on UCI datasets

4.2 Rule Performance Metrics Quantifying the benefits of EOP models is difficult since no generally accepted objective metric of interpretability is available to satisfy intuition of an arbitrary user. Nevertheless, since EOP model consists of a set of rules, we can rely on the existing rule evaluation criteria (devising appropriate metrics is a formidable challenge beyond the scope of this paper). We have chosen four metrics based on selection criteria and recommendations provided in comprehensive surveys by Geng and Hamilton [12] and Lenca et al. [19] and designed to scoring rules $A \rightarrow B$. Bayes Factor (BF) and Lift (L) are simple metrics of high intelligibility, that have been shown to perform well at identifying relevant rules [19]. Additionally, we consider Normalized Mutual Information (NMI) for the properties described in [12] and because of its applicability to hierarchical models. Further, we use J-Score (J), a well-studied symmetric measure of interestingness that considers impact of positive and negative examples in data. The exact formulas used to compute these metrics are shown below.

$$BF(A \rightarrow B) \quad = \quad \frac{p(A|B)}{p(A|\overline{B})} = \frac{n_{AB} n_{\overline{B}}}{n_B n_{A\overline{B}}}$$

$$
\begin{aligned}
L(A \to B) &= \frac{p(B|A)}{p(B)} = \frac{n \cdot n_{AB}}{n_A n_B} \\
J(A \to B) &= p(A)\Big(p(B|A)\log\frac{p(B|A)}{p(B)} \\
&\quad +(1-p(B|A))\log\frac{1-p(B|A)}{1-p(B)}\Big) \\
A &= \wedge_{i=1}^{d} a_i \\
NMI(A \to B) &= \frac{\left(\sum_{i=1}^{d} p(a_i,b)\log_2\frac{p(a_i,B)}{p(a_i)p(b)}\right)}{-\sum_{i=1}^{d} p(a_i)\log_2 p(a_i)}
\end{aligned}
$$

These metrics are designed to score individual rules, but they can be adapted to scoring rule sets by computing an average weighted by the rules' frequencies of use. This technique enables a valid comparison between EOP and CART since both produce sets of mutually exclusive rules. In the formula below - detailing the model evaluation for EOP - $\mathcal{M}$ denotes the model, D is the depth of the hierarchy, R_i represents the set of regions that are handled by classifier h_i. R_i has cardinality q_i. $R_i(x)$ denotes the event that a point x belongs to a region in R_i, while $C(h_i, x)$ denotes the event that h_i correctly classifies point x.

$$
\begin{aligned}
\mathcal{M} &= \{(R_i = \{r_1 \dots r_{q_i}\}, h_i) \mid i = \overline{1,D}\} \\
&= \{\cup(A_i \to B_i) \mid i = \overline{1,D}\} \\
A_i &= \left(\wedge_{j=1}^{i-1} \neg R_j(x)\right) \wedge R_i(x) \qquad B_i = C(h_i, x)
\end{aligned}
$$

The metric M for the model is computed as a linear combination of component Ms obtained for individual levels of the hierarchy that are visited during prediction, weighted by their corresponding support:

$$
M(\mathcal{M}) = \sum_{i=1}^{D} p(A_i) M(A_i \to B_i)
$$

Tables 3 and 4 summarize explainability scores of EOP and CART models obtained using previously described synthetic and real-world data sets. For synthetic data, we computed means, standard deviations and p-values from paired T-test, to determine statistical significance. Bayes Factor becomes numerically unstable whenever one of the components of the hierarchical model is fully homogeneous with respect to the output class distribution. It is reflected in the result tables with symbol "Inf", and we ignore the corresponding datasets in computing summary scores. The empirical results show that EOP regularly identifies more explainable regions of the feature space, according to all metrics but J-Score. The difference in J-Scores observed on synthetic data does not appear statistically significant, and J-Score results for real-world data are mixed.

4.3 Case study of EOP Models: Cell Data Explainability is useful in many practical applications. It

Table 3: Metrics for CART and N_EOP - with decision stumps - on artificial data

	CART				EOP			
	BF	L	J	NMI	BF	L	J	NMI
DS1	Inf	0.005	0.223	0.018	3.109	0.016	0.262	0.440
DS2	Inf	0.010	0.236	0.052	1.818	0.048	0.136	1.093
DS3	1.160	0.014	0.267	0.014	1.372	0.019	0.009	0.337
DS4	1.620	0.004	0.005	0.048	1.498	0.038	0.400	0.559
DS5	1.454	0.008	0.113	0.062	2.826	0.027	0.146	0.703
DS6	1.445	0.007	0.148	0.041	1.719	0.013	0.096	0.785
DS7	4.181	0.008	0.195	0.033	4.875	0.027	0.265	0.854
DS8	Inf	0.010	0.236	0.052	1.818	0.048	0.136	1.093
DS9	Inf	0.008	0.198	0.051	2.143	0.024	0.670	0.369
Mean	*1.972*	*0.008*	*0.180*	*0.041*	*2.458*	*0.029*	*0.235*	*0.693*
Stdev	*1.340*	*0.003*	*0.081*	*0.016*	*1.398*	*0.013*	*0.199*	*0.289*
tTest	*0.012*	*0.001*	*0.252*	*0.000*				

Table 4: Metrics for CART and N_EOP on real data

	CART				EOP			
	BF	L	J	NMI	BF	L	J	NMI
MB	1.982	0.004	0.389	0.040	1.889	0.007	0.201	0.502
BCW	1.057	0.007	0.004	0.011	2.204	0.069	0.150	0.635
BT	0.000	0.009	0.210	0.000	Inf	0.021	0.088	0.643
V	Inf	0.020	0.210	-0.010	2.166	0.040	0.177	0.383
Mean	*1.520*	*0.010*	*0.203*	*0.010*	*2.047*	*0.034*	*0.154*	*0.541*

is often the case in scientific research when understanding of the results is as important as discovering patterns. The goal of one such application is to determine whether a stem cell has been subjected to a treatment. The hope is that it could be determined using a set of measurements taken under a microscope, such as the area and perimeter of the cell, the stage of the cell cycle at the time of the observation, the generation the cell belongs to, as well as some other measurements. Figure 6 shows the EOP model obtained after training on 5,000 data points evaluated on an equally large test set. In this case, the hierarchical model only identifies one-dimensional intervals in which data can be confidently discriminated, rather than multidimensional combinations. This behavior can be tuned using the EOP dimensionality regularization parameter λ.

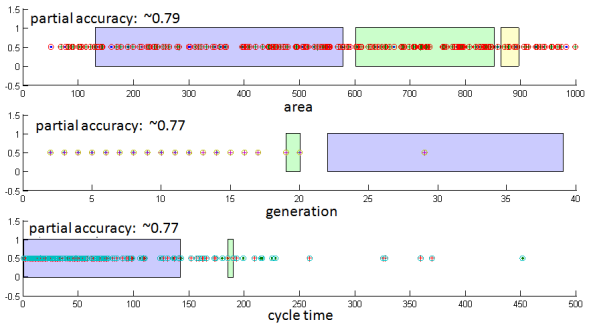
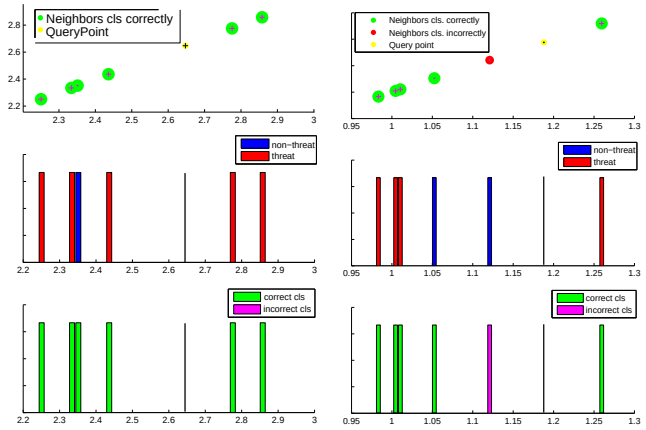


Figure 6: Explanatory projections for the Cell dataset

The interpretation is that cells with specific features

falling within learned intervals can be safely classified as having been subjected to treatment. If, going from the top level of the hierarchy, classification by cell area is inconclusive; cell generation and cycle time are considered. Intuitively, for small or very large cells it is more difficult to determine whether treatment was applied. However, falling back on generation and then cycle time helps to provide a confident answer in many such cases. Note that the overall classification accuracy of EOP on this data is 77%, comparing favorably to 72% obtained with a random forest model.

4.4 Explaining Nuclear Threat Detection Next, we show how EOP can assist operators of a nuclear Threat Detection System (TDS). Our TDS analyses measurements of radiation and contextual features obtained from vehicles crossing the border - a total of 25 real-valued features. The operators can decide to allow or deny passage, or subject the vehicle to a time consuming detailed inspection. N_EOP produces a model of depth five, where each projection uses a single feature of the data. Additional queries are in the Appendix.



(a) TDS Query 1 (b) TDS Query 2

Figure 7: TDS queries for the N-EOP model

Figure 7 displays two queries presented to the TDS system using the N_EOP model. For each query, the top graph displays the query point - yellow - and its neighbors from the training set - green, if classified correctly and red if classified wrong. Threats are marked with '+' while non-threats with '.' - true labels for the neighbors, assigned label for the query. The center graph shows the true empirical class density of the neighboring points. This plot is useful since multiple neighbors may have the same feature value and will overlap on the plot. The bottom graph represents the empirical density of the classifier's accuracy. Well-classified data is shown in green and misclassified data points in magenta. The graphs are aligned, with the x-axis tracing the feature with which the query point is being classified. For the top graphs, the y-axis is simply a duplicate of the x-axis intended to provide additional spacing. Query 1 is classified as a threat, its correctly classified neighbors are also threats; likely, it would not need further investigation to be denied border crossing. Query 2 is classified as a non-threat. However, it is equally distanced from a threat training instance and a misclassified non-threat. It should be subjected to further inspection. Out of the two query points, one looks suspicious and indeed, Q2 is incorrectly classified. Through this display of results, the operator may avoid a hazard thanks to the explanatory capability of EOP.

4.5 Robustness It is often useful in practice to identify subsets of data that are hard to confidently classify and set them aside. A variant of the EOP - Accuracy Targeting EOP, ATEOP - screens all projections for the largest robust regions where data can be classified with acceptable accuracy. Unlike the standard EOP implementation, ATEOP does not dynamically lower the error threshold to handle all data if possible. Instead, it aims to maintain overall reliability of classification, and ignores left-overs that are hard to deal with.

ATEOP is given a parameter ϵ, representing the allowable classification error rate, and an α, representing the minimum support of data that regions must provide in order to be included in the model. Regions that meet these criteria on training data are verified using a separate validation set. If validation turns out too restrictive, ϵ is gradually reduced to force more robust selections during training. If multiple regions meet the threshold criteria, the one with the most extensive data coverage is chosen. The data in that region is then removed, and the process continues until all data is processed or no new satisfactory regions can be found.

Figure 8 displays the trade-off between achieving the required accuracy and data coverage. We ran ATEOP on one of the synthetic data sets explained above. In the graph we plot the obtained accuracy measured on the data included in the model (which is as high as required), as a function of the accuracy threshold ϵ. We also plot accuracy of the default classifier applied to the left-over data. As ϵ increases, ATEOP is allowed to become more error-tolerant, the less data has to be left out, and the accuracy based on data included in the model goes down. Eventually, when accuracy threshold is lenient enough to allow all data to be included in some part of the ATEOP hierarchy, the two plots converge. For reference, we also plot the accuracy achieved by CART. It is as expected slightly higher than the accuracy achieved at the convergence of the two ATEOP characteristics. A desired balance,

which varies by application, can be obtained through cross-validated selection of the threshold.

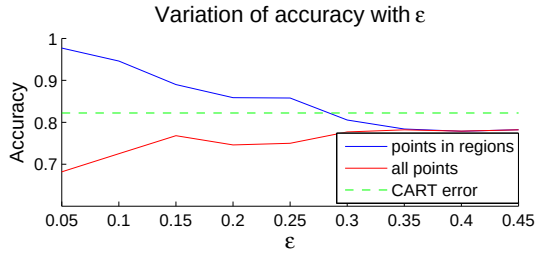


Figure 8: Accuracy of ATEOP as a function of the allowable classification error rate: Accuracy for data covered by the model (blue); Accuracy for all data (red); Accuracy of the CART model on all data(green dashed)

5 Conclusions

We have introduced Explanation-Oriented Partitioning, an algorithm that learns explainable classifications. It works by identifying high confidence regions in low-dimensional projections where data is easy to classify.

These regions can be used as contextual explanations to accompany predictions made for test queries. EOP can incorporate any externally provided classifiers. It relies on these to identify interesting projections of data that form a hierarchical, low-complexity model, that maintains competitive predictive accuracy while providing compact models when compared to relevant peers. The most important outcome however is that EOP classification results are easy to understand by human users. We have shown parametric and non-parametric variants of the procedure for identification of explainable regions of feature space.

The presented algorithm is shown to closely match the performance of boosting while providing transparent models. It also fares well when compared to alternative approaches by producing more compact models for little or no loss in accuracy. EOP algorithms are capable of finding expressive projections of data while maintaining high fidelity. The compact resulting models capture the essence of data and offer intuition to the users.

References

[1] L. Breiman. Bagging predictors, 1996.

[2] L. Breiman. Random forests, 2001.

[3] L. Breiman. Statistical modeling: The two cultures. *Statistical Science*, 2001.

[4] Leo Breiman, Jerome Friedman, Charles J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks, 1995.

[6] Mark W. Craven and Jude W. Shavlik. Extracting Tree-Structured Representations of Trained Networks, 1996.

[7] Pedro Domingos. Knowledge discovery via multiple models. *Intelligent Data Analysis*, 2:187–202, 1998.

[8] Eulanda M. Dos Santos, Robert Sabourin, and Patrick Maupin. A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. *Pattern Recogn.*, 2008.

[9] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[10] Yoav Freund. Boosting a weak learning algorithm by majority, 1995.

[11] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, 1997.

[12] Liqiang Geng and Howard J. Hamilton. Interestingness measures for data mining: A survey. *ACM Comput. Surv.*, 2006.

[13] Tin Kam Ho. The random subspace method for constructing decision forests. *Pattern Analysis and Machine Intelligence, IEEE Trans on*, 1998.

[14] Breiman L. Stacked regressions. *Machine Learning*, 1996.

[15] Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm, 1988.

[16] Bing Liu, Minqing Hu, and Wynne Hsu. Intuitive representation of decision trees using general rules and exceptions, 2000.

[17] Michael Mampaey, Nikolaj Tatti, and Jilles Vreeken. Tell me what i need to know: succinctly summarizing data with itemsets, 2011.

[18] Michael J. Pazzani, Subramani Mani, and W. Rodman Shankle. Beyond concise and colorful: Learning intelligible rules, 1997.

[19] Patrick Meyer B Phillipe Lenca, Benot Vaillant A, and Stphane Lallich C. On selecting interestingness measures for association rules: user oriented description and multiple criteria decision aid, 2008.

[20] Kelvin Sim, Ardian Kristanto Poernomo, and Vivekanand Gopalkrishnan. Mining actionable subspace clusters in sequential data, 2010.

[21] Peter Sollich and Anders Krogh. Learning with ensembles: How over-fitting can be useful, 1996.

[22] Kai Ting, Jonathan Wells, Swee Tan, Shyh Teng, and Geoffrey Webb. Feature-subspace aggregating: ensembles for stable and unstable learners. *Machine Learning*, 2011.

[23] G.I. Webb and Z. Zheng. Multistrategy ensemble learning: reducing error by combining ensemble learning techniques. *Knowledge and Data Engineering, IEEE Trans on*, 2004.

[24] Leland Wilkinson, Anushka Anand, and Dang N. Tuan. CHIRP: a new classifier based on composite hypercubes on iterated random projections, 2011.

[25] David H. Wolpert. Stacked generalization. *Neural Networks*, 1992.

Explanation-Oriented Classification via Subspace Partitioning - Supplementary Material

Madalina Fiterau* mfiterau@cs.cmu.edu
Artur Dubrawski† awd@cs.cmu.edu

October 16, 2012

1 EOP vs CART on Artificial Data

Our comparison of EOP to CART uses the same artificial data as the boosting comparison. The left side of Table 1 shows how nonparametric EOP fares against CART on synthetic data. The results are obtained through cross-validation. While CART is on average about 3% more accurate, EOP uses models that are considerably less complex in terms of the number of weighted decisions - the weights are equal to the number of data dimensions used by decisions. The differences in performance and model complexity are significant in terms of the paired T-test.

Table 1: Comparisons of nonparametric EOP and parametric EOP - with decision stumps - against CART in terms of accuracy and complexity on artificial data

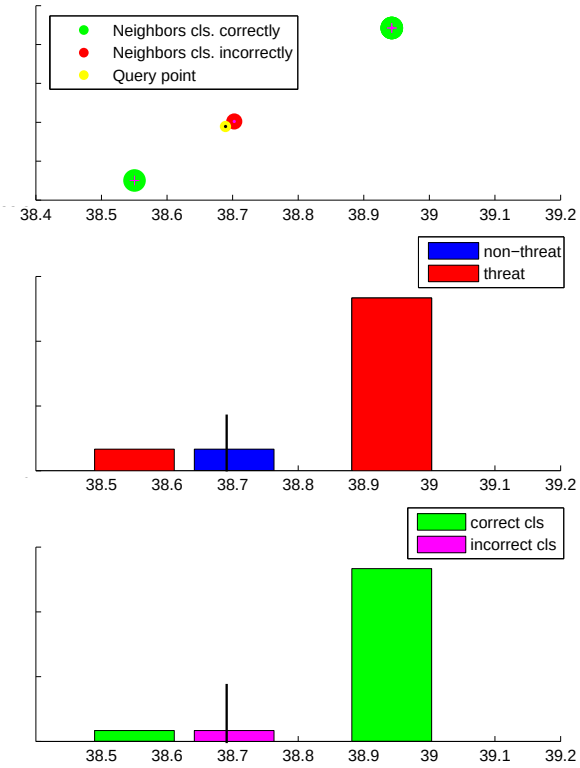
	Cart Acc	Cart Comp	eop_N Acc	eop_N Comp	Cart Acc	Cart Comp	eop_P Acc	eop_P Comp
1	0.947	21	0.924	16	0.85	11	0.837	2
2	0.918	15	0.903	7	0.82	9	0.747	4
3	0.951	27	0.93	8	0.826	17	0.741	3
4	0.838	41	0.754	5	0.914	9	0.79	6
5	0.927	31	0.887	9	0.842	11	0.838	4
6	0.873	33	0.83	6	0.884	5	0.886	3
7	0.88	27	0.828	15	0.874	7	0.747	2
8	0.957	27	0.923	24	0.834	5	0.753	3
9	0.934	33	0.891	7	0.84	9	0.705	5
10	0.959	17	0.936	11	0.812	25	0.693	3
μ	*0.918*	*27.2*	*0.881*	*10.8*	*0.85*	*10.8*	*0.773*	*3.3*
σ	*0.041*	*7.91*	*0.059*	*5.92*	*0.031*	*6.07*	*0.062*	*1.49*
p	*12E-5*	*63E-5*			*7E-4*	*2E-3*		

The right side of Table 1 summarizes the performance of the parametric version of EOP - learning axis-aligned rectangular regions - as compared to CART on a different set of data. Although the accuracy is, on average, less than that of CART, there still are some datasets for which EOP_P performs better. Importantly, the parametric models are considerably less complex, and rely on easy to interpret regions. Assignment of a testing data point to the appropriate region can be done using two vector comparison operations per tried region, so parametric EOP's complexity is proportional to the expected number of rectangles against which a query needs to be tested. Note that the results for CART differ between these two tables due to randomness of the data generation process.

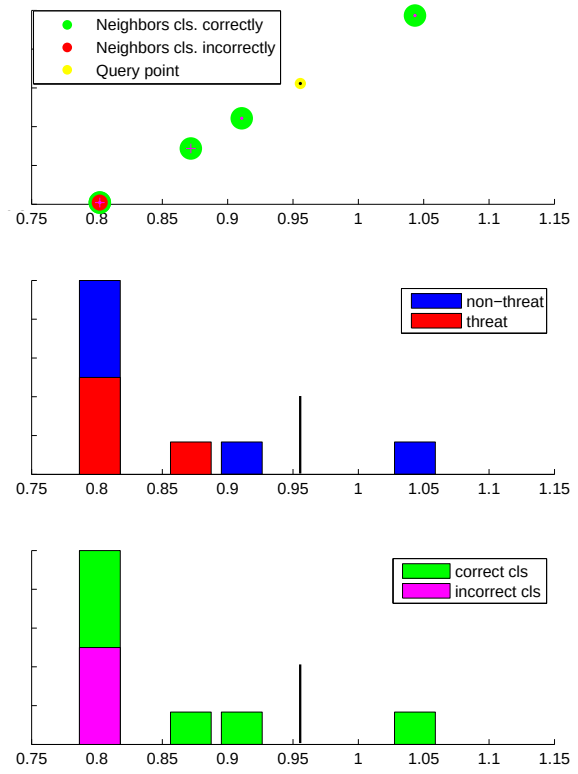
2 More Nuclear Threat Detection Queries

Figure 1 displays two other queries presented to the TDS system using the N_EOP model. For each query, the top graph displays the query point - yellow - and its neighbors from the training set - green, if classified correctly and red if classified wrong. Threats are marked with '+' while non-threats with '.' - true labels for the neighbors, assigned label for the query. The center graph shows the true empirical class density of the neighboring points. This plot is useful since multiple neighbors may have the same feature value and will overlap on the plot. The bottom graph represents the empirical density of the classifier's accuracy. Well-classified data is shown in green and misclassified data points in magenta. The graphs are aligned, with the x-axis tracing the feature with which the query point is being classified. Q3 is classified as a non-threat, with the majority of distant neighbors accurately classified as threats and the nearest neighbor misclassified as a threat, making it a candidate for further inspection. The Q4 is labeled as a non-threat; its closest neighbors are also correctly classified non-threats, and more distant neighbors are correctly classified threats. Only the most distant is misclassified, which makes it an unlikely candidate for inspection. It is clear that the classification of Q4 is more convincing than that of Q3, and indeed Q4 is a correct prediction while Q3 is wrong. This shows how the information provided by EOP can be used to pinpoint unreliable predictions.

*Machine Learning Department, Carnegie Mellon University
†Robotics Institute, Carnegie Mellon University



(a) TDS Query 3



(b) TDS Query 4

Figure 1: TDS queries for the N-EOP model

3 Case study of EOP Models: Spam Data

Another case study involves a spam detection problem - we use the Spambase dataset from the UCI repository. The data contains about 4,000 records and 57 features. EOP obtains a spam prediction accuracy of 80%, with the top three projections and the associated high confidence regions shown in 2. Each two-dimensional EOP region is depicted with a distinct color. The scatter plots show testing data resolved at subsequent levels of the EOP hierarchy.

The top-level classifier labels everything as spam. The high confidence region, which indeed does enclose mostly spam test examples, does not have a high incidence of the word 'your', but it shows a high incidence of capital letters, which makes an intuitive sense. The classifier in the next iteration is less likely to mark something as spam; the selected regions immediately reflect this semantic change: the threshold for the incidence of the word 'your' is lowered and the required incidence of capitals is increased. The square region on the left also encloses examples marked as 'not spam' because of the lower incidence of capitals correlated with the appearance of the pronoun.

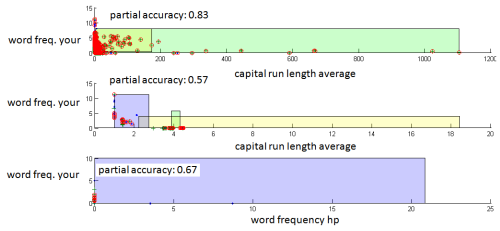


Figure 2: Explanatory projections for Spambase

4 Pattern Identification

A separate experiment illustrates the ability of EOP to identify patterns in data. Additional relatively small clusters of synthetic one class data from Gaussian distributions were injected into randomly chosen dimensions of the data.

The first column in the table in figure 3 shows which sets of features were impacted by the injections. This experiment involved 7 simultaneous injections containing about the same number of points. The columns of the table correspond to levels of the EOP hierarchy. Each cell i, j of the table shows how many of the injected data points belonging to pattern i have been captured by some region at iteration j. The darker the color, the more points have been explained. Results show that EOP selects relevant projections of data at early iterations, to quickly reveal the injected overdensities - it deals with many of the datapoints at the very first iteration. Subsequently, the second projection explains another batch of points - corresponding to patterns 4,5

	Number of points picked at each iteration			
Pattern Features	Iteration 1	Iteration 2	Iteration 3	Default
[1,10]	31	5	3	1
[2,7]	36	5	4	0
[5,6]	44	8	1	0
[7,6]	34	10	5	0
[9,2,1]	29	10	1	0
[6,9,4]	21	12	4	0
[1,10,3,5]	41	7	3	0

Figure 3: Illustration of how EOP deals with injected lower-dimensional patterns - number of points from each pattern explained at each stage.

and 6. The first row of the table corresponds to a pattern spanning features 1 and 10. It consists of 40 data points. 31 of them were handled at level 1, the following 5 at level 2, and 3 at level 3 and 1 was left for the default classifier.

5 Pseudocode

Algorithm 2 shows the pseudocode for the EOP algorithm. The pseudocode for the AT-EOP algorithm is displayed in Algorithm 1.

Algorithm 1 ATEOP Algorithm

```
ATEOP(ϵ,α)
ϵ0 = ϵ; Classifiers=[]; Regions=[]
while TrainData is not Empty and foundProjection
do
  foundProjection = false
  while ϵ0 > 0 and not foundProjection do
    minRegionSize = α*size(TrainData)
    for all π ∈ Π do
      [h,R] = ObtainClassifierAndRegions(TrainData,ϵ0)
      PointsInSet = R.filterPoints(CalibrationData)
      CalibrationError =
      h.classificationError(PointsInSet)
      if CalibrationError< ϵ
          and PointsInSet.size()>minRegionSize
      then
        minRegionSize = PointsInSet.size()
        Classifiers.add(h); Regions.add(R)
        foundClassifier=true
      end if
    end for
    if foundProjection then
      TrainingData.eliminatePointsIn(R)
      CalibrationData.eliminatePointsIn(R)
    end if
  end while
end while
```

Algorithm 2 EOP Algorithm

```
EOP(trainingData, calibrationData, ϵ, λ, η)
Classifiers = []; Regions = []
while trainingData is not empty do
  (h, π) = SelectClassifier(trainingData, λ, η)
  trError = (Classify(h,trainingData.π)
          ≠ currentData.label)
  calibrationError = (Classify(calibrationData.π)
          ≠ calibrationData.label)
  sets = ObtainSets(trainingData,trError,ϵ)
  FilterSets(sets,calibrationData,calibrationError)
  if pointsInRegion is Empty then
    increase(ϵ)
  else
    Classifiers.append(h); Regions.append(sets)
    currentData.remove(PointsInRegion(trainingData,sets))
  end if
end while
return (Regions,Classifiers)
```

```
SelectClassifier(data, λ, η)
Π = CombineFeatures(data)
for all π ∈ Π do
  h.append(TrainClassifier(π,λ)); pred =
  Classify(h,π)
  score.append(avg(pred ≠ data.label) + η*size(π))
end for
idxBest = index(score,min[score])
return (h[idxBest],Π[idxBest])
```